This listing of claims will replace all prior versions and listings of claims in the application:

Listing of claims:

1-90. (Canceled)

91. (Currently amended) A load module embodied on a computer-readable medium, the load module comprising:

a load module header including a public portion and a private portion;

said public portion including identification information and information describing at least one aspect of a hardware or software platform on which said load module is designed to execute;

said private portion including at least one correlation tag including information used to determine whether a method has authorization to call or load the load module; and

a load module body, including:

executable programming specifying that information relating to a use of the load module be communicated to a remote site; and

a reference to data, at least some of said data being associated with or used by said executable programming.

92. (Previously presented) The load module of Claim 91, in which said at least one aspect includes the level or degree of security present or available on such platform.

93. (Currently amended) The load module of Claim 91, in which said at least one aspect includes ~~thea~~ type of computer.

94. (Currently amended) The load module of Claim 91, in which said at least one aspect includes ~~the~~a type of software running on such platform.

95. (Previously presented) The load module of Claim 91, in which said at least one aspect includes one or more computer languages recognized by said platform.

96. (Currently amended) An operating system embodied on a computer-readable medium, comprising:
component assembling programming which assembles a plurality of elements into a component, said component assembling programming including;
(a) validation programming used to validate said elements, said validation programming including:
(1) tag checking programming used to check the identity, validity or integrity of elements by comparing tags incorporated in said elements to expected values; and
(2) element identification and referencing programming; and
(b) communications programming used to communicate at least one result of said tag comparison to a remote site; and
an object switch which controls and communicates objects, said object switch including:
one or more stream interfaces; and
a container manager used to manage secure containers.

97. (Currently amended) The operating system of Claim 96, in which:
said operating system is designed to operate correctly with applications programs written to run on one or more versions of a conventional~~the Microsoft Windows~~ operating system.

98. (Previously presented) The operating system of Claim 96, in which:
said operating system runs in a processing environment; and

said operating system includes at least one added component delivered at some point after the initial installation of said operating system at said processing environment.

99. (Previously presented) The operating system of Claim 98, in which:
said added component provides scalability to said operating system.

100. (Previously presented) The operating system of Claim 98, in which:
said added component comprises a component assembly made up of a plurality of elements.

101. (Previously presented) The operating system of Claim 96, said operating system further comprising:
channel definition programming which sets up and initializes channels in which component assemblies are assembled.

102. (Previously presented) The operating system of Claim 96, in which:
said component assembling program includes programming which checks said components for information regarding the manner in which said components are designed to be assembled into a component assembly,
said programming requiring that said components only be assembled in the manner specified by said information.

103. (Currently amended) The operating system of Claim 96, in which:
said tag checking programming includes comparison programming which compares the contents of ~~the~~a public tag associated with an element with the contents of a private tag associated with that element.

104. (Previously presented) The operating system of Claim 103, in which:
said comparison programming includes programming which decrypts said private tag

prior to said comparison.

105. (Previously presented) The operating system of Claim 96, in which:
said tag checking programming includes comparison programming which compares the contents of a tag associated with an element with the contents of a tag associated with a process requesting said element.

106. (Previously presented) The operating system of Claim 105, in which:
said comparison programming includes programming which decrypts said tag associated with said element prior to said comparison.

107. (Previously presented) The operating system of Claim 96, in which:
said tag checking programming includes comparison programming which compares the contents of a tag associated with an element with the contents of a tag stored in a secure processing unit;
said comparison designed to determine whether said tag associated with said element is the same as the tag most recently assigned to said element by said secure processing unit.

108. (Previously presented) The operating system of Claim 96, further comprising:
e-mail management programming.

109. (Previously presented) The operating system of Claim 108, in which:
said e-mail management programming includes programming which recognizes and controls secure e-mail or secure e-mail attachments.

110. (Previously presented) The operating system of Claim 109, in which:
said e-mail management programming includes programming which routes secure e-

mail or secure e-mail attachments to a secure memory location.

111. (Previously presented) The operating system of Claim 96, further comprising:
an object repository manager.

112. (Previously presented) The operating system of Claim 111, in which:
said object repository manager provides services relating to access to an object repository.

113. (Previously presented) The operating system of Claim 96, in which:
said validation programming includes certificate programming which checks digital certificates associated with said elements.

114. (Previously presented) The operating system of Claim 113, in which:
said certificate programming includes programming which compares an expiration date on at least some of said digital certificates with the current date.

115. (Previously presented) The operating system of Claim 113, in which:
said certificate programming includes programming which extracts one or more keys from at least one of said digital certificates and uses said one or more keys to decrypt information associated with the digital certificate from which said one or more keys was extracted.

116. (Previously presented) The operating system of Claim 96, in which:
said object switch includes a stream router which includes programming which routes streams to and from said stream interfaces.

117. (Previously presented) The operating system of Claim 96, in which:

said one or more stream interfaces include at least one real time stream interface.

118. (Previously presented) The operating system of Claim 117, in which:
said real time stream interface includes programming designed to accept and route real time data stream information.

119. (Previously presented) The operating system of Claim 101, in which:
said channels further serve to pass events to methods and load modules specified to process the events.

120. (Previously presented) The operating system of Claim 96, in which:
said component assembling programming includes programming which uses a blueprint in said component assembly process.

121. (Currently amended) A component assembly embodied on a computer readable medium, comprising:
a first load module and a second load module, each load module comprising:
a load module header, made up of a public portion and a private portion;
said public portion including identification information and information describing at least one aspect of a hardware or software platform on which said load module is designed to execute;
said private portion including at least one correlation tag including information used to determine whether a method has authorization to call or load the load module; and
a load module body, including:
executable programming; and
a reference to data, at least some of said data being associated with or used by said executable programming,
said first load module executable programming including programming requiring

the storage of audit information relating to use of the component assembly.

122. (Previously presented) The component assembly of Claim 121, in which said at least one aspect includes the level or degree of security present or available on such platform.

123. (Currently amended) The component assembly of Claim 121, in which said at least one aspect includes ~~the~~a type of computer.

124. (Previously presented) The component assembly of Claim 121, in which said at least one aspect includes the type of software running on such platform.

125. (Previously presented) The component assembly of Claim 121, in which said at least one aspect includes one or more computer languages recognized by said platform.

126. (Currently amended) A component assembly embodied on a computer readable medium, comprising:

a first load module and a second load module, each load module comprising:

a load module header, made up of a public portion and a private portion;

said public portion including identification information;

said private portion including at least one correlation tag and information on the stack size used by or required by said load module, said correlation tag including information used to determine whether a method has authorization to call or load the load module; and

a load module body, including:

executable programming; and

a reference to data, at least some of said data being associated with or used by said executable programming,

said first load module executable programming including programming requiring the storage of information uniquely identifying a device at which said component assembly is stored.

127. (Currently amended) A component assembly embodied on a computer readable medium, comprising:

a first load module and a second load module, each load module comprising:

a load module header, made up of a public portion and a private portion;

said public portion including identification information;

said private portion including at least one correlation tag, and an access tag, said access tag being made up of at least two fields, each of which can be accessed and used separately and said correlation tag including information used to determine whether a method has authorization to call or load the load module; and

a load module body, including:

executable programming; and

a reference to data, at least some of said data being associated with or used by said executable programming,

said first load module executable programming including programming requiring communicating a unique identification for a device at which said component assembly is stored to a remote location.

128. (Currently amended) A computer processing system comprising:

a processing unit operable to execute computer programming, wherein the computer programming comprises:

a component assembler which assembles a plurality of elements into a component assembly, said plurality of elements each including at least one tag, said component assembler including a validator that validates each of said plurality of elements, said validator including a tag checker that checks at least one of: (a) the identity, (b) the validity ~~and~~or (c) the integrity, of said plurality of elements by comparing

said tags incorporated in said plurality of elements to expected values; and

an object switch coupled to said component assembler, said object switch including:

(a) a stream router that communicates component assemblies;

(b) one or more stream interfaces coupled to said stream router;

(c) a container manager that, in use, manages said component assemblies; and

(d) an object switch interface that interfaces said object switch with said component assembler; and

a communications module which communicates a unique identifier of the computer processing system or a user of the computer processing system to a remote location.